DAY 02 OF HACKATHON

PLANNING THE TECHNICAL FOUNDATION

SHOP.CO

Marketplace : General E-commerce

Technical Requirements

1.Frontend Requirements (Next.js):

- Design user-friendly pages for browsing products.
- Responsive layout using Next.js themes and Tailwind CSS.
- Essential pages:
    - Home
    - Product Listing (using Next.js)
    - Product Details
    - Cart and Checkout
    - Order Confirmation.

Backend Requirements (Sanity):

- Use Sanity for managing product listings, customer data, and orders.
- Install essential plugins for:

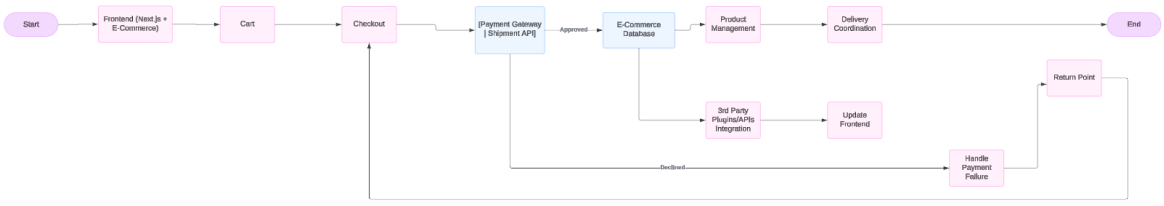
    - Payment gateway integration (e.g., PayPal, Stripe).
    - Shipment tracking (e.g., Shipengine).
- Customize the Next.js database to include product details, orders, and customer records.

Third-Party Integrations:

- Payment Gateways: E-Commerce extensions for secure transactions.
- Shipment Tracking: Integrate tracking plugins for real-time updates.
- APIs: Use REST APIs for connecting E-Commerce data with external platforms.

2. System Architecture:

High-Level Architecture Diagram:



Data Flow:

1. User visits the marketplace frontend and browses products.
2. API fetches product data from the Next.js database.
3. User adds items to the cart and proceeds to checkout.
4. Payment processed via payment gateway plugin.
5. Order and shipment data are saved in the database and updated via APIs.

3. Key Workflows:

1. User Registration:
    a. Flow: User signs up -> Data saved in the Next.js database -> Confirmation email sent via Mailchimp/SMTP plugin.
2. Product Browsing:
    a. Flow: User views categories -> Data fetched dynamically using E-Commerce REST API -> Products displayed with sorting and filtering options.
3. Order Placement:
    a. Flow: User adds items to the cart -> Proceeds to checkout -> Order details saved via E-Commerce.
4. Shipment Tracking:
    a. Flow: Tracking updates fetched via Shipengine -> Real-time status displayed on the user account page.

4. Plan API Requirements:

Example Endpoints Using E-Commerce REST API:

Endpoints	Method	Description	Example Response
/next-json/ec/v3/products	GET	Fetch all product details	{ "id": 1, "name": "shirt", "price": 50 }
/next-json/ec/v3/orders	POST	Create a new order	{ "orderId": 123, "status": "Processing" }
/next-json/ec/v3/zones	GET	Fetch available shipping zones	{ "zoneId": 1, "name": "Local Delivery", "methods": [{ "id": "free_shipping" }] }
/next-json/ec/v3/orders/{id}	GET	Fetch order details by ID	{ "orderId": 123, "status":

			"Shipped", "ETA": "3 days" }

5. Sanity Equivalent with Next.js:

In Next.js, data management is handled via Custom Post Types (CPTs) and E-Commerce. Below is an equivalent schema for your marketplace:

- Products:
  - Fields: Name, Price, Stock, Description, Image.
  - Plugin: E-Commerce.
- Orders:
  - Fields: Customer Info, Products Ordered, Payment Status.
  - Plugin: E-Commerce.
- Customers:
  - Fields: Name, Email, Address.
  - Plugin: E-Commerce.